Docker 技术在天文数据档案库系统测试中的应用*

刘英[1,3]　黄茂海[1,2,3]　张墨[1,3]
（1.　中国科学院国家天文台，北京 100101；　2. 中国科学院大学天文与空间科学学院，北京 100049；
3.　中国科学院空间天文与技术重点实验室，北京 100101）

摘要：随着 Docker 技术的快速发展和越来越多被用于天文软件应用程序的部署，如何理解容器技术，将容器技术运用到天文软件开发测试工作中，成为每一个相关从业人员应该思考和快速掌握的核心虚拟化技术。文章通过对 Docker 技术在天文数据档案库系统测试中的应用，实现对科学数据产品长期存档与数据产品查询检索，支持各级数据产品、工程数据、标定数据、辅助数据的管理，测试科学数据产品存储、检索、提取、维护、分析、控制功能，指出在传统软件环境部署及测试中，天文软件环境复杂且运行时依赖较多第三方库支持，测试中耗费大量时间定位软件缺陷重现测试环境，介绍 Docker 技术在天文数据档案库系统测试应用中带来的优势和重要性，实现测试环境标准化、测试数据隔离性、测试功能扩展性，提高测试工作效率。同时也为容器技术在其它天文软件测试与应用提供借鉴参考。

关键词：Docker 技术；软件测试；轻量级；标准化；扩展性；天文数据档案库系统

中法合作天文卫星（SVOM）的研究对象是来自宇宙深处的伽玛射线暴发，伽玛暴是宇宙中极端明亮的瞬变源，产生于大质量恒星塌缩或致密双星并合时恒星级黑洞或磁星喷射出的在视线方向的极端相对论性喷流【1】。伽玛暴爆发时，在很小的空间里释放了巨大的能量，如伽玛暴和引力波暴等瞬变源光学对应体的后随观测与搜索对研究瞬变源的物理属性有重要作用【2】。伽玛暴研究具有拓宽或根本改变我们对一些重大天体物理问题的认识的潜力。SVOM 通过对科学目标的实现，为研究暂现源涉及的从恒星、星系到宇宙学等天体物理学中的多个领域研究、解决相关天体物理学领域中的若干重大问题提供观测和数据基础【1】。

天文数据档案库系统是 SVOM 中方科学中心重要组成部分，实现科学数据产品长期存档与数据产品查询检索，支持各级数据产品、工程数据、标定数据、辅助数据等的管理，数据产品版本可控，保证数据安全性与完整性。数据档案系统提供统一的用户及数据存取管理平台，实现跨平台对科学数据产品存储、检索、提取、维护、分析、控制功能【3】。

作为国际合作项目，SVOM 天文数据档案库系统需要在中法双方不同单位和场景进行测试集成调试。与一般基于互联网的应用部署不同，天文软件一般在部署运行时需要非常多的第三方库支持，运行时也有诸多参数以满足不同要求。天文领域对容器技术的使用还停留在天文应用软件的封装层面【4】，如何让天文学家快捷方便的使用天文数据档案库系统，让测试人员在中法双方调试环境中，不会因服务器迁移而进行重新部署测试环境的重复劳动，针对国际合作项目中服务器需要迁移的工作需求，即在中法双方调试环境中运行，我们可以采用将容器镜像导入到法方服务器的方式，启动需要的容器服务，降低部署过程中出现问题的风险，节约时间和人力成本。Docker 技术具有轻量级，易移植性且保证环境一致性的特点，为天文数据档案库系统环境部署和测试的首选解决方案。

1. Docker 技术简介

* 基金项目：国家自然科学基金 (11603049) 资助。

Docker 是基于 Go 语言实现的云开源项目，其核心解决的问题是利用 LXC 来实现类似 VM 的功能，用更加节省的硬件资源提供给用户更多的计算资源。 Docker 的主要目标是“Build, Ship and Run Any App, Anywhere”，通过对应用程序的封装（Packaging）、分发（Distribution）、部署（Deployment）、运行（Runtime）等生命周期的管理，真正实现“一次封装，随处运行”【5】。

Docker 容器技术具有标准化、轻量级、迁移性和扩展性等显著特点，为了更好的理解容器技术，这里需要了解两个非常重要的基本概念：

镜像（Image）：是类似虚拟机的镜像，可以将它理解为一个面向 Docker 引擎的只读模板，包含文件系统【5】。镜像可以从 Docker Hub 公共镜像源中下载，也可以在本地自行创建镜像。

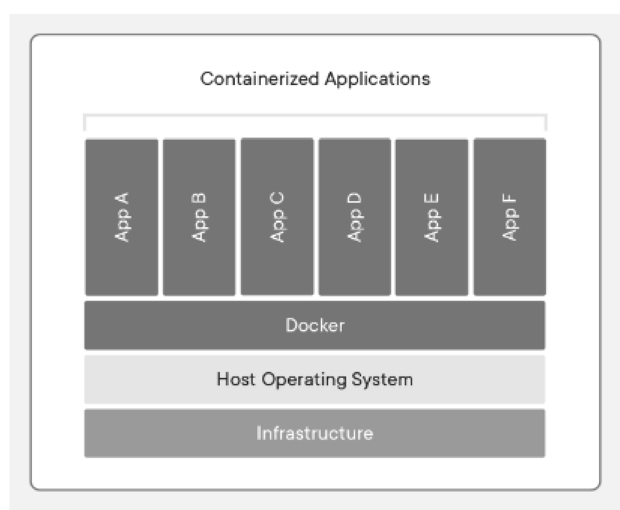
容器（Container）：是运行起来的镜像，所有容器都是运行在镜像之上，如果我们把镜像理解为程序，那么容器就可以看作是进程。

2. Docker 与 VM 在软件测试中的应用特点及区别

在软件测试中，软件环境的搭建直接影响测试结果的准确性和真实性。很多软件测试人员都有使用虚拟机(VM)的经验，即在传统物理机的基础上克隆出虚拟机，这种镜像具有真实 Windows, Linux 操作系统的功能，在此基础上安装软件，配置参数，解决软件测试环境搭建和资源紧张的难题。

Docker 技术快速部署，高效构建应用，与宿主机共享内核资源的优势，使测试人员在保证测试环境的标准性，数据隔离性，和缺陷环境重现中得到很好的应用。特别是秒级的快速启动 Docker 容器服务，软件测试效率显著提高。

Docker 相对于 VM，在轻量级、资源利用率方面具有明显的优势。我们可以从图 1【6】的对比看到，由于容器技术是虚拟化操作系统而不是硬件，是应用程序的抽象，多个容器可以在同一个机器上运行，与其他容器共享操作系统，占用空间少，可以处理更多的应用程序，从而减少对操作系统的需求。虚拟机是物理硬件的抽象，每个 VM 都包含操作系统、应用程序及必要的二进制文件和库【6】。



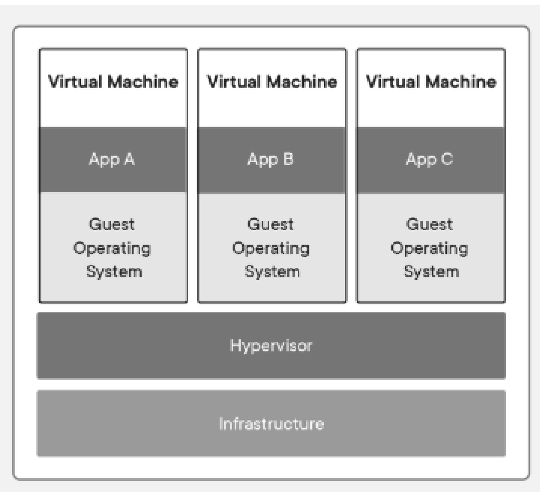


图 1 Docker Vs VM

3. Docker 技术在天文数据档案库系统测试中的应用

3.1 天文数据档案库项目简介

天文数据档案库主要实现科学数据产品长期存档、数据产品查询、检索、提取及分析等功能，支持跨平台运行，主要提供用户管理接口、数据注入接口、数据检索接口以及数据获取接口。软件运行支持 Linux 平台，在本次项目测试中服务器端操作系统使用 Ubuntu, 数据库使用 MySQL, Web 服务器使用 Tomcat，客户端运行使用平台 HIPE（Herschel Interactive Processing Environment）即欧洲空间局最复杂的大型空间项目赫歇尔空间天文台的地面系统软件平台【7】。

针对项目的实际运行情况，不仅需要满足数据处理功能要求，系统的长期可扩展性，以及后期大系统集成测试，和上天设备的同步运行等各种复杂因素综合考虑，如果采用传统的服务器客户端部署系统环境测试，势必造成每一次变动都需要重新部署测试环境，耗费大量的时间和人力成本。

SVOM 天文数据档案库系统作为中法双方合作项目，不同科学家常常会使用不同运用软件开发语言版本，比如在中方运行环境使用科学数据检索 Python2 的脚本没有错误,在法方部署环境中运行法国科学家创建的科学数据检索 Python3 的脚本出现抛错，虽然我们可以采用工具如 Python Virtual Environment 或者 Anaconda 的安装去帮助解决 Python 版本不同的问题，但是安装工具 Anaconda 中本身碰到的错误，如选择是否添加到系统路径还是使用默认 Python 的路径不正确，会直接导致环境使用 Python 运用出错。如果有一种更为快捷的方式帮助测试人员快速进行环境配置，准确定位发现软件缺陷，将会使测试工作进行更加顺畅。

采用 Docker 技术，在软件测试中保证复杂的天文软件环境一致，重现测试中发现的软件缺陷并复制测试环境，有效运用有限的测试环境资源和人力资源，极大提高测试工作效率。

3.2 天文数据档案库环境搭建

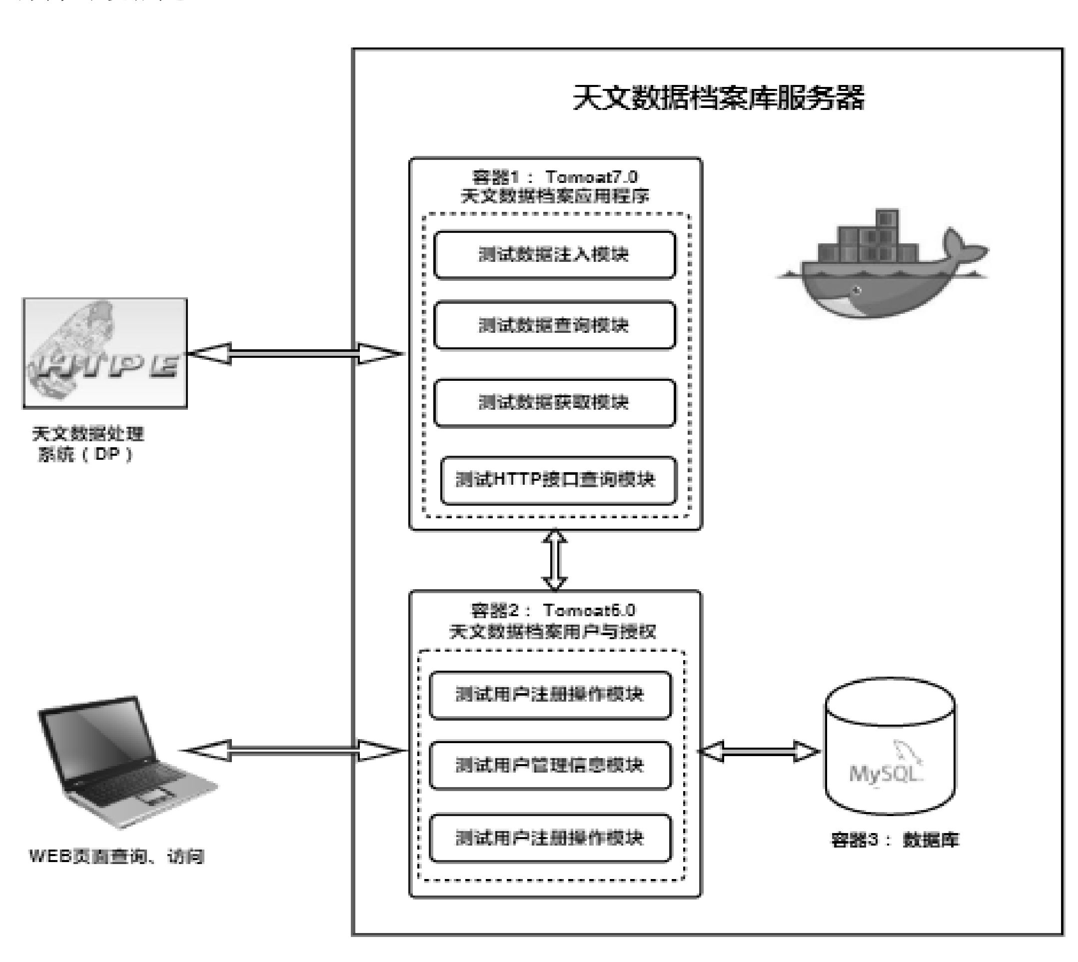


图 2 天文数据档案库测试模块及环境部署图

通过图 2，我们可以看到 Dock 技术如何在实际项目测试环境搭建中的应用：

1） 首先建立一个宿主机环境，本项目使用 Ubuntu 14.04.5 并在该环境安装 Docker 软件包和依赖包，

2） 建立数据库容器，下载 MySQL 5.7 镜像，使用命令

```
docker pull mysql/mysql-server:5.7.24
```

启动容器并指定端口 3306，将数据库容器的端口映射到宿主机上，使用命令

```
docker run -p 3306:3306 --name mysql -e MYSQL_ROOT_PASSWORD=123456 -d mysql
```

成功启动数据库容器并进入容器，将数据库配置文件 my.cnf 中 port 修改为容器启动的端口 3306，执行 sql 语句创建用户管理库及创建权限库。

3） 建立天文数据档案库应用程序容器，拉取 Tomcat-7.0.67 容器镜像，使用命令

```
docker pull tomcat:7.0.67
```

启动容器并指定端口 8000，使用命令

```
docker run -p 8000:8000 tomcat:7.0.67
```

运行容器，将数据档案库应用程序 war 包复制到该容器对应路径，并进行相关参数以及和数据库的配置，输入正确的数据库容器 ip 和数据库密码，重新启动容器使用命令

```
docker stop <containerID>
docker start <containerID>
```

我们可以通过 curl 命令在应用程序容器里检查天文数据档案库应用程序容器和数据库容器连接是否成功

curl http://ip:3306

4） 建立用户管理子系统容器，拉取 Tomcat-6.0.45 容器镜像并指定端口启动容器，使用命令

```
docker pull tomcat:6.0.45
```

成功启动并进入容器，对相应配置文件和数据库进行配置，重启容器，检查数据注入是否成功

5） 建立天文数据档案库系统镜像，对数据库容器、天文数据档案库应用程序容器、用户管理子系统容器创建新的 Docker 镜像，使用命令

```
docker commit -a “tester” -m “test” <containerID> test:v01
```

完成镜像创建后，通过 docker images 查看新创建镜像是否已经提交在本地。

6） 导出导入镜像，针对天文数据档案库系统需要在中法双方调试的实际需求，我们可以通过导出在中方调试成功的镜像，导入到法方运行环境中，从而避免部署过程中出现问题的风险，使测试人员更加关注软件功能测试。导出镜像使用命令

```
docker save -o test.tar test:v01
```

导入镜像使用命令

```
docker load --input test.tar
```

7） 天文数据档案库系统中载荷数据处理存储解决方案，提供 API 接口依赖于 HIPE【7】提供的 ProductStorage Java API 接口【3】，实现天文数据档案库在 HIPE 中交互访问，在 HIPE 脚本指令区，通过脚本调用产品归档、

检索、获取、分析功能，在 HIPE 控制台区，可以查看脚本运行的结果，即通过数据注入接口和数据查询接口成功将数据产品存入天文数据档案库并查询出已存入的科学数据产品。

8） 通过图 3，我们可以看到 Docker 技术在天文数据档案库系统测试中的应用即容器创建、运行、提交镜像的全流程，将数据库、应用程序、用户管理子系统分别部署在不同的 3 个容器；对用户管理子系统和应用程序的各个模块进行测试，验证科学数据产品存储、检索、提取、维护、分析、控制功能是否正常工作；将测试通过的容器镜像提交到容器仓库。

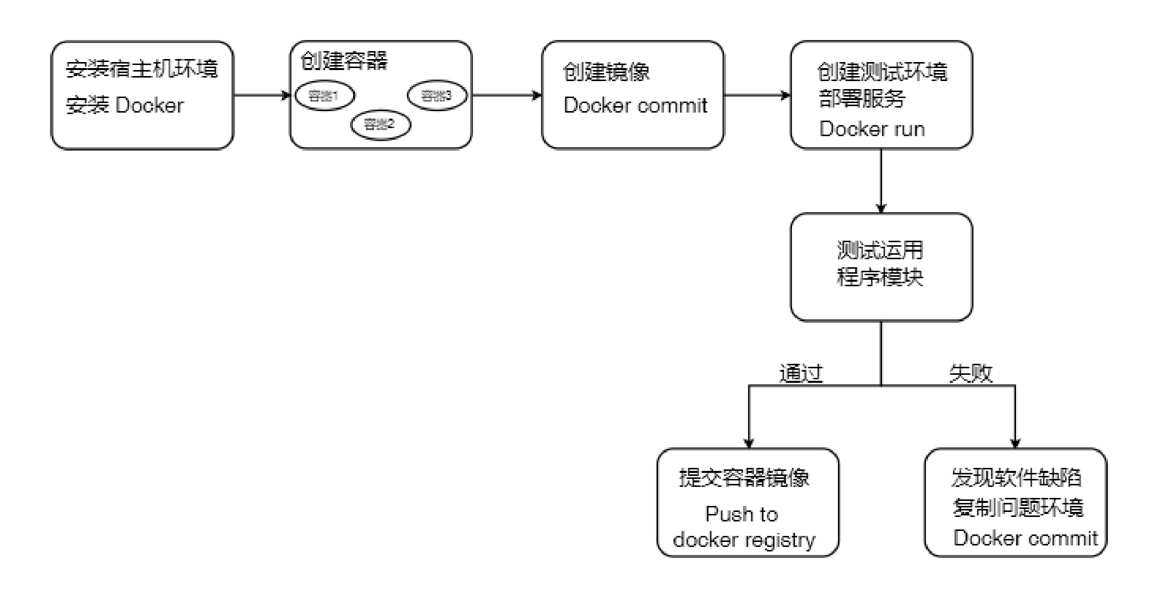


图 3　天文数据档案库系统测试流程图

4 Docker 技术在天文数据档案库系统测试中的优势

测试人员都有这样的经验，同样的环境配置和产品发布版本，在不同的机器或者不同的集成环境里安装就有可能失败，造成这样的结果有很多因素，而环境的标准化无疑是避免这种问题的有效解决方式。通过以上项目实例，Docker 技术在天文数据档案库系统测试中的主要优势有以下几点：

1） 测试环境标准化：在传统测试工作中，不同操作系统和数据库版本有不同限制，比如 Linux 平台中 Ubuntu 和 Centos,运行安装命令就完全不同,Ubuntu 安装命令执行 apt-get,Centos 安装命令执行 yum，在天文数据档案库系统测试中，因为 MySQL 版本的不同，SQL 语句执行 datetime 字段出现高版本 5.7 运行低版本 SQL 语句抛错，数据不能导入的错误。

天文数据档案库系统测试环境选用 Ubuntu14 和 MySQL5.7,在交付用户使用环境采用 Docker 容器将操作系统和数据库打包到新的镜像，容器标准化使开发、测试、运维环境保持一致【8】，避免用户由于使用不同版本操作系统或数据库而造成运用程序抛错，在需要 debug 的时候，开发测试人员可以直接使用打包镜像针对问题研究，而不被环境问题牵扯损耗时间。

2） 测试安装简洁化：软件测试过程中，安装测试是测试工作的第一步，也是软件功能、集成、压力测试的重要保障。在传统测试过程中，以天文数据档案库系统为例，由于项目设计需要安装不同版本 Web 服务器 Tomcat，在设置环境参数 CATALINA_HOME 不注意区分多版本共存的问题，极易导致启动一个 Tomcat 而另一个版本的 Tomcat 在启动。

选择 Docker 技术，在同一个宿主机环境里，运行两个不同版本 Tomcat 容器，在每个容器里配置对应的环境参数，通过不同的端口映射到宿主机上，分别启动两个容器,环境隔离有效避免了因环境参数配置的问题，导致启动一个 Tomcat 而另外一个启动的错误。安装测试时间减少，安装测试步骤更加简化，卸载安装时只需要删除容器，在几秒钟之内启动一个新的基础容器进行环境配置，快速解决环境卸载不干净的问题。

3） 测试功能扩展性：天文数据档案库系统采用迭代开发模式，通过用户使用反馈逐步完善产品功能，传统测试过程中，每次发布新的应用程序版本或产品功能修改更新，测试人员都需要重新搭建测试环境，重复劳动造成测试工作效率降低。

运用 Docker 技术，我们可以把 Tomcat6.0/7.0 作为基础镜像,每次发布新的运用程序版本，在基础镜像的基础上更新配置快速搭建新环境，通过对各个独立容器进行扩展实现无缝化伸缩，而不需要对整个运用程序进行重新开发。同时在 SVOM VOEvent 网络系统中，也成功应用 Docker 技术部署 VOEvent 原型系统，对 SVOM 中方科学中心的复杂项目集成测试工作流程和效率带来极大提高【9】。

4） 测试数据隔离性：传统软件测试过程中，常常因为测试数据的增加，特别是天文软件环境配置复杂，天文数据档案库系统需要多次采用不同类型和不同数据量的数据进行测试，将各个子系统和数据库都部署在一个环境，极易造成测试中产生脏数据，重复读取数据的问题。

采用将数据库、应用程序、用户管理子系统分别部署在不同的容器，实现各自独立，测试数据分离，测试环境干净，可以随时切换测试场景，方便定位软件缺陷。同时，数据库容器独立，需要卸载数据库，通过容器删除命令就可以简单实现，避免卸载数据库环境不干净的问题。

5） 问题重现便捷性：软件测试的一个重要目的就是尽可能多的发现软件中的缺陷。作为测试人员，发现缺陷后重现问题环境是帮助推进修正软件缺陷的重要工作环节，即使将代码和数据同时发布，比如一些未做文档记录的假设项、依赖项、配置项都会使重现问题非常困难【10】。传统测试方法过程中，由于测试数据交叉测试案例复杂，发现软件缺陷常常无法将测试环境共享给开发团队。如果用传统 VM 快照技术复制测试环境包含操作系统极易造成资源占用过大【11】。

对于本项目测试中碰到的 SQL 语句版本问题导致数据不能导入数据库，因为需要对数据库本身的调试和对天文数据档案库的集成调试，找到建立用户系统不成功的根源，数据库、应用程序、用户管理子系统几个容器都需要保证问题环境完全相同，通过容器镜像可以在几秒钟之内把测试环境成功复制，容器共享宿主机操作系统，层级简化，磁盘空间占用少。在试验中 Docker1.4.1 的环境新建一个容器占用空间需要 12 Kilobytes 磁盘空间，而同样的测试环境用 VM 做快照占用资源将达到超过一百个 Megabytes【12】。

软件测试特点	传统测试方法存在问题	运用 Dock 技术解决方案	优势对比
测试环境标准化	选择不同的操作系统和数据库版本，造成开发、测试、用户环境出现不同问题	通过打包镜像，用户使用环境、开发 debug 环境和测试环境完全一致	开发、测试、运维环境保持一致，保证配置文件、运行路径、权限等配置的标准化
测试安装简洁化	对多版本共存的环境，易造成参数设置错误，不同版本	创建不同版本 Tomcat 容器，映射不同端口到宿主机，实	安装测试时间减少，步骤简化，环境隔离避免共存环境

	运用运行冲突的问题	现多版本 Tomcat 准确启动	参数设置和运用启动冲突
测试功能扩展性	对新功能增加，新版本软件发布，需要重新搭建测试环境，重复劳动多	通过创建基础镜像，新版本软件测试在基础镜像上搭建新环境	减少搭建测试环境的不必要重复劳动，节约时间和人力成本，提高测试工作效率
测试数据隔离性	在复杂的天文软件环境，测试数据增加，易造成产生脏数据，重复读数据的问题	将数据库、应用程序、用户管理子系统分别部署在不同的容器	测试数据分离，测试环境干净，测试容器独立，可以随时切换测试场景
问题重现便捷性	未做文档记录的假设项、依赖项、配置项使重现问题非常困难	通过容器镜像，在几秒钟之内复制测试环境，快速定位问题	容器共享宿主机操作系统，资源成本占用少,使重现问题环境更加容易

表 1　天文数据档案库系统容器技术应用和传统测试方法对比优势

5 总结

从表 1 可以看到，Docker 化服务器端环境并应用于软件测试中，相比于传统测试方法，可以极大节省测试人员部署调试环境的时间，使测试环境标准化、测试数据隔离性、重现问题环境更加便捷。同时各个系统部署在各自独立的容器，进行配置连接，形成统一的整体应用程序，Docker 技术的这种高扩展性对于可以将应用程序独立于各个容器设计的项目尤为适用。在整个项目测试过程中，同时运行多个 Docker 容器，对 CPU、磁盘、内存等资源占用率小，启动和备份相对于传统虚拟机技术速度显著加快。Docker 技术的应用，为测试环境的稳定性和标准化提供了快捷高效的解决方案，为软件测试工作效率提高和成本降低提供了保障。

致谢:感谢刘飞飞老师对项目测试过程中的技术建议。感谢 SVOM 项目组工程人员在项目测试和系统搭建过程中给予的建议和帮助。

参考文献:

[1] A 类先导专项实施方案-SVOM 科学应用系统二期

[2] 符夏川，吴潮，黄茂海，张墨，卢晓猛. 瞬变源快速自动响应观测系统的研究与实现［J］. 天文研究与技术，2017,14(4)：452.

[3] 光学载荷数据处理存储解决方案-用户手册

[4] 姚坤，戴伟，杨秋萍，梅盈，石聪明，王锋. 基于容器技术的天文应用软件自动部署方法［J］. 天文研究与技术，2019,16(3)：321.

[5] 杨宝华，戴王剑，曹亚仑. Docker 技术入门与实战. 2015,1：3-4，2:9.

[6] Docker. https://www.docker.com/resources/what-container

[7] HIPE: Ott, S. 2010, ASP Conference Series, 434, 139

[8] D. Morris, S. Voutsinas, N.C. Hambly, R.G. Mann. : Use of Docker for deployment and testing of astronomy software［J］. Astronomy and Computing, Volume 20, July 2017: 105.

[9] Mo Zhang, Maohai Huang, Chao Wu :Prototype VOEvent Network Systems based on VTP and XMPP for the SVOM Chinese Science Center. SpaceOps 2016 Conference (AIAA), Daejeon, Korea;

doi:10.2514/6.2016-2316

[10] Boettiger, C.: An introduction to docker for reproducible research. ACM SIGOPS Oper. Syst. Rev. 49(1), 71 - 79 (2015)

[11] Dav Clark, Aaron Culich, Brian Hamlin, Ryan Lovettr.: BCE: Berkeley’s Common Scientific Compute Environment for Research and Education [C] //Proceedings of the 13th Python in Science Conference (SciPy 2014).(2014).

[12] Karl Matthias, Sean P. Kane. Docker: Up and Running. First Edition. Printed in the United States of America: O’Reilly Media, Inc., 2015: 27-28.